SBMT–STEINER BACKUP MULTICAST TREE

Wu-Hsiao Hsu[†], Jenhui Chen[*], Shiann-Tsong Sheu[‡], and Chih-Feng Chao[§]

[†]Department of Computer Science and Information Engineering
Ming Chuan University, Tao-Yuan, Taiwan 333, R.O.C.
wuhsiao@mcu.edu.tw
[*]Department of Computer Science and Information Engineering
Chang Gung University, Tao-Yuan, Taiwan 333, R.O.C.
jhchen@mail.cgu.edu.tw
[‡]Department of Electrical Engineering
Tamkang University, Tamsui, Taiwan 251, R.O.C.
stsheu@ee.tku.edu.tw
[§]Department of Computer Science and Information Engineering
Tamkang University, Tamsui, Taiwan 251, R.O.C.
cfchao@giga.net.tw

2004/1/15

Abstract

In the backbone network, a Steiner multicast tree (SMT) will be established for multicast members to minimize the traffic load on networks. However, a communication link or node may fail due to some accidental factors during the transmission period. Downstream nodes with respect to the failed link/node will be forced to leave this tree. In order to guarantee the quality of service (QoS), it is desirable to have some schemes for the multicast tree so that such termination of service can be avoided or at least, reduced. In this paper, we propose a fixed SMT algorithm (FSA) to construct the Steiner backup multicast tree (SBMT). Based on FSA, for each 'critical' path, an alternate route with enough bandwidth will be reserved such that most fatal failures in the network can be recovered immediately. The way to determine critical paths is based on statistical analysis. In addition, an adaptive SMT algorithm (ASA) is proposed to construct both SMT and SBMT on unreliable networks. The adjustment of the SBMT when nodes dynamically join or leave the SMT is also discussed. The degree of fault tolerance of the proposed strategies is evaluated and compared by simulation. Simulation results demonstrate that FSA and ASA improve the reliability in stable and unstable networks, respectively. Moreover, the dynamic joining process of a node will be sped up by taking both SMT and SBMT into considerations. Simulation results are presented to demonstrate the effectiveness of the optimization.

[*]Corresponding Author: Department of Computer Science and Information Engineering, Chang Gung University, 259 Wen-Hwa 1st Road, Kwei-Shan, Tao-Yuan, Taiwan 333, R.O.C. Phone: +886-3-211-8800-5990, Fax: +886-3-211-8700. Email: jhchen@mail.cgu.edu.tw

Key Words: algorithm, quality of service, fault tolerance, multicast, reliability.

1 Introduction

Many multimedia systems, such as video conferencing systems, need dedicated bandwidth to guarantee real-time communications. Asynchronous transfer mode (ATM) networks are considered to provide the most suitable transport technique due to their ability to flexibly support a wide range of services with quality of service (QoS) guarantees. Similarly, one can also use the resource reservation protocol (RSVP) [1] or differentiated services (DiffServ) [2] to reserve bandwidth. RSVP runs over the network layer (e.g., IPv4 or IPv6) and uses the existing multicast tree to reserve bandwidth. Receivers choose the level of resources reserved and are responsible for initiating and keeping the reservation active as long as they want to receive the data. In DiffServ, traffic is differentiated into a set of classes for scalability, and network nodes provide priority-based treatment according to these classes.

Actually, a communication link or node may fail due to some accidental factors during the connection holding time. When a link (node) fails, downstream nodes with respect to this link (node) will be excluded from the multicast tree. For the sake of guaranteeing the QoS, it is necessary to re-establish new connections with enough bandwidth for these nodes right away. To do this, broadcast messages will be generated to search for restoration paths one by one. This approach to restore a failed link is known as the dynamic method. However, the remaining bandwidth on the different route(s) in the network may be insufficient to support these connections at the moment the failure occurs. Besides, it will take a considerable time to recover all failures. Thus, each working path should pre-arrange its backup routes with an appropriate amount of bandwidth for the purpose of fault tolerance. Such an approach is known as the preplanned method. It is important for networks to provide high-reliability services such as, for example, emergency rescue applications and tactical communications. Of course, the reserved bandwidth will be wasted if all links or nodes work well. In a word, it is trade off between the efficiency and reliability. It is desirable to design a scheme to provide the best quality of fault tolerance but reserving the minimum bandwidth.

1.1 Backup Multicast Tree

Many new applications require point-to-multipoint (multicast) communications, which send the same packet to a group of destinations. In order to consume a minimal amount of resources for such applications, a minimal cost-spanning tree is applied. A common solution is to model the network as a graph $G = (V, E)$ where the nodes represent routers or switches and the edges represent links between nodes, and to generate a multicast tree that includes the source and destination nodes. In a point-to-point (unicast) network, we are interested in finding the shortest path between a pair of nodes so that the propagation delay of each

packet is minimized. On the other hand, in the multicast problem, we are interested in the minimum cost subtree or multicast tree which connects all destination nodes and the source node. Finding this minimum cost subtree, which is known as Steiner multicast tree (SMT) [3] in graphs, is a complicated problem. It has been shown that deriving the SMT is NP-complete [4].

Previous work on a multicast tree can be classified into two categories. The first one assumes that all destination nodes are unchanged during the transmission period. That is, one only focuses on finding a heuristic algorithm to minimize the tree's cost so that the derived cost is close to the minimum cost. Several polynomial-time heuristic algorithms [5, 6, 7] have shown that the derived cost is twice the optimum in the worse case. Secondly, the multicast members are allowed to leave or join the multicast tree during the transmission period. Two famous heuristic algorithms, the greedy algorithm [11] and the weighted greedy algorithm (WGA) [5], are proposed to solve the dynamic problem. Besides, there is some work describing alternative routing schemes which ensure that the network restores itself under any single link failure [8, 9, 10].

In this paper, we not only consider link by link or path by path, but also consider a whole tree in network topology. Besides, none of the previous work considers the operations of node joining and leaving. We focus on how to protect an established SMT. Intuitively, constructing another independent multicast tree will provide the best reliability. However, a considerable amount of network bandwidth will be wasted also. Therefore, we do not focus on developing a heuristic algorithm for SMT but rather on finding a suitable Steiner backup multicast tree (SBMT) from an established SMT. In other words, we will deal with the problem of how to find the SBMT with the minimum bandwidth in order to derive the maximum degree of reliability. A statistical approach is employed to determine the important edges (paths), denoted as critical edges (paths) in this paper, in the SMT which are the candidates to be protected. Because the multicast members may dynamically join or leave, it is desirable to have a scheme to quickly modify the SBMT to respond to the change. In this paper, we also propose an efficient strategy to adjust the SBMT to accommodate the changed SMT. In the proposed strategy, both SBMT and SMT are used to assist finding a best attachment node when a node is joining. This strategy not only saves much computation time for a node's joining but also results in an acceptable total cost.

In an unreliable network, lots of edges/nodes in SMT may fail during the service holding time. As the SBMT quickly takes over the failure path one by one, the cost of the recovery tree will become higher and higher. Therefore, the way of finding the SMT and SBMT over an unreliable network should be different from the traditional approach. The cost difference between SMT and SBMT should be as small as possible. In this paper, we propose another enhancement algorithm to construct the SMT and SBMT for unreliable networks.

The rest of this paper is organized as follows. Section 2 addresses the characteristics

of SBMT. Section 3 introduces the proposed algorithm for constructing the Steiner backup multicast tree (SBMT). Section 4 introduces the strategy for a node's dynamic joining and leaving. We also describe the enhancement algorithm for establishing both SMT and SBMT in unreliable networks. Second 5 shows the simulation model and results. Some conclusions are given in Section 6.

2 Definitions and Analysis

2.1 Second Best Steiner Multicast Tree

Before proceeding, we formally define the graph terminology and SMT problem. Let $G = (V, E)$ be a graph with a finite set of vertices V and edge set E. Also, a cost function, $w : E \rightarrow Z^+$ (where Z^+ stands for the set of positive real numbers excluding 0), assigns a positive integer number (cost) to each edge in G. For simplicity, we assume the graph is undirected. That is, (u, v) and (v, u) are the same edge and $w(u, v) = w(v, u)$. In this paper, the terms vertex (edge) and node (link) are used interchangeably. The degree of a vertex is the number of edges incident on it. A path from vertex u to vertex v is a sequence of vertices and edges $u, (u, v_1), (v_1, v_2), \cdots, (v_k, v), v$ and is denoted as $u \xrightarrow{p} v$. A bridge is defined as an edge whose removal disconnects G. We also define an undirected graph to be a *connected* graph if every pair of vertices is connected by at least one path.

Definition 1 *Given a set $D \subseteq V$, the SMT problem is to find a tree T that contains all vertices in D such that the total cost $w(T)$ is minimal. That is,*

$$\sum_{e \in T, e \in E} w(e) \quad \textit{is minimal.}$$

Basically, protecting a constructed SMT means finding a different multicast tree with enough pre-reserved bandwidth for the edges excluded from SMT. These edges are denoted as 'backup' or 'standby' edges. A good backup multicast tree should not only maximize the number of backup edges but also minimize the amount of reserved bandwidth. However, it is hard to achieve these two goals at the same time. For example, if an algorithm is designed to find a tree with the maximum number of backup edges, the maximum amount of bandwidth will be wasted and the call blocking probability will become higher accordingly. Briefly speaking, from the user's point of view, the maximal degree of connection reliability should be provided during the call holding time; but from the network point of view, the network performance will suffer from obtaining a larger call blocking probability. In fact, it is meaningless to protect all edges in a tree. That is, the main concern of devising the backup multicast tree for a practical network is how to minimize reserved bandwidth rather than how to completely provide backup links for all edges. In this paper, the proposed algorithm of constructing the backup multicast tree starts by finding a tree of the second minimal cost;

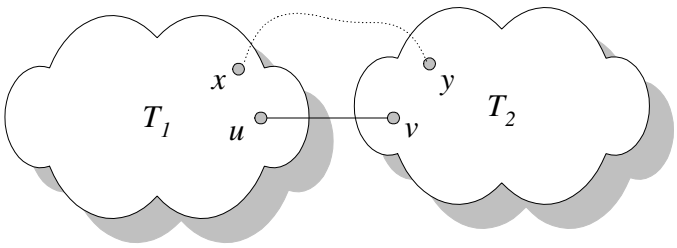


Figure 1: The SBS is derived by replacing edge (u, v) with path $x \xrightarrow{p} y$.

and then, the tree is adjusted step by step until the reliability criterion is satisfied. The second best SMT (SBS) problem is defined as follows.

Definition 2 *Given a set $D \subseteq V$, the SBS problem (SBSP) is to find a tree T that contains all vertices in D such that the cost difference between $w(T)$ and $w(SMT)$ is minimal and $\exists e \in T$, $e \notin SMT$. That is,*

$$\sum_{e \in T-SMT, e \in E} w(e) - \sum_{e' \in SMT-T, e' \in E} w(e') \quad \textit{is minimal.}$$

In this paper, we will show that the problem of finding a *SBS* is still very difficult. Before introducing the NP-completeness of the SBS problem (SBSP), we first introduce a crucial NP-complete problem: minimum edge SMT problem (MESP) as follows:

Definition 3 *Given a set $D \subseteq V$ and a positive integer k, the MESP is to find a tree T that contains all the vertices in D such that the number of edges in T is less than or equal to k.*

It has been shown that the MESP is NP-complete [12]. Now, we use the following theorem to show the NP-completeness of SBSP.

Theorem 1 *Given an undirected graph $G = (V, E)$, a multicast member set $D \subseteq V$ and a positive integer k, the problem of determining whether there exists a multicast tree $T = (V_T, E_T)$ such that $D \subseteq V_T$ and $|E_T| = k$ is NP-complete.*

Proof. We prove this theorem by showing that the SBSP has the same complexity as the MESP. Consider the simplest version of SBSP that assumes each edge has the same cost c in the graph. Let the total cost of the SMT T' be C. We know there are C/c edges in T'. We know the second best SMT, say T, has a larger total cost than T'. If we let $k = C/c + 1$ (i.e., $\sum_{e \in T} w(e) - \sum_{e' \in T'} w(e') = c$), the simplest SBSP is reduced as the MESP. Since the MESP is NP-complete, we know the simplest SBSP is also NP-complete. As a result, we conclude that the SBSP is NP-complete.

Because solving SBSP is difficult, we propose a heuristic algorithm to find the SBS. Before describing how to derive the SBS, we first introduce a method to determine whether there exists another multicast tree that differs from SMT. Let T denote the obtained SMT.

It is obvious that each edge in T is a bridge. If an edge, say (u, v), is removed from T, it decomposes T into two subtrees T_1 and T_2 accordingly. For simplicity, let V_1 and V_2 denote the sets of vertices of T_1 and T_2, respectively. We have $u \in V_1$ and $v \in V_2$. It is clear that a different multicast tree will be derived if there exists at least one alternative path $x \xrightarrow{p} y$ $(x \in V_1, y \in V_2)$, which connects these two subtrees T_1 and T_2 as shown in Fig. 1. Notice that nodes x and y may be the same as nodes u and v, respectively. Let $S_e(T)$ denote the set of all available alternative paths for an edge e in T. Therefore, the set, denoted as $S(T)$, contains all possible paths for constructing different multicast trees from T and can be obtained by the following equation:

$$S(T) = \bigcup_{e \in T} S_e(T). \tag{1}$$

Clearly, if $S(T) = \phi$, no other multicast tree can be found.

Lemma 1 *Let $G = (V, E)$ be an undirected connected graph with cost function $w : E \rightarrow Z^+$ and suppose that $|E| \geq |V|$. Let T be a SMT of G for a multicast group. Assume set $S(T)$ is not empty. Let $x \xrightarrow{p} y$ be the path of the minimum cost in $S(T)$ and $x \xrightarrow{p} y \in S_{(u,v)}(T)$. Then, the tree $T - \{(u, v)\} \bigcup \{x \xrightarrow{p} y\}$ is the SBS of G.*

Proof. Recall that any path in $S_e(T)$ connects two subtrees T_1 and T_2, which are decomposed by removing an edge e in T. Therefore, if a path $x \xrightarrow{p} y$ of the minimum cost c in $S_{(u,v)}(T)$ is added into T, it will form a cycle in T. That is, $u \xrightarrow{p} x \xrightarrow{p} y \xrightarrow{p} v \xrightarrow{p} u$. If we remove the edge (u, v) from T, another multicast tree T' in G will be derived. The total cost of this new multicast tree is $w(T') = w(T) - w(u, v) + w(x \xrightarrow{p} y)$. Because T is SMT, we have $w(u, v) \leq w(x \xrightarrow{p} y)$. In other words, $w(T) \leq w(T')$ and, thus, T' is the found SBS.

Usually, the selected alternate path of edge (u, v) is the loop-free second-shortest path from u to v. It is important to note that there is no cycle and only one path exists from source to each vertex in D of the SMT [11]. The same property should be maintained during construction of the SBS. Obviously, the derived SBS is one of the SBMTs for a given SMT. For simplicity, in the following descriptions of algorithms, we consider the SBS as the desired SBMT. Because most of the edges of SBS overlap those of SMT, the SBMT (i.e., SBS) provides low reliability. A simplex example of constructing SMT and SBMT is given in Fig. 2. The number above each link is the cost of the link. Although the cost difference is minimal (=1), there are five overlapping edges. This is quite unsafe for SMT even though this SBMT is established. Therefore, we adjust (remove) the overlapping edges between SMT and SBMT one by one. The decision to remove an overlapping edge depends on that edge's importance.

2.2 Critical Edge

Now we deal with the problem of having too many overlapping edges between SMT and SBMT. The crux of the problem is how to choose a proper overlapping edge to replace at each

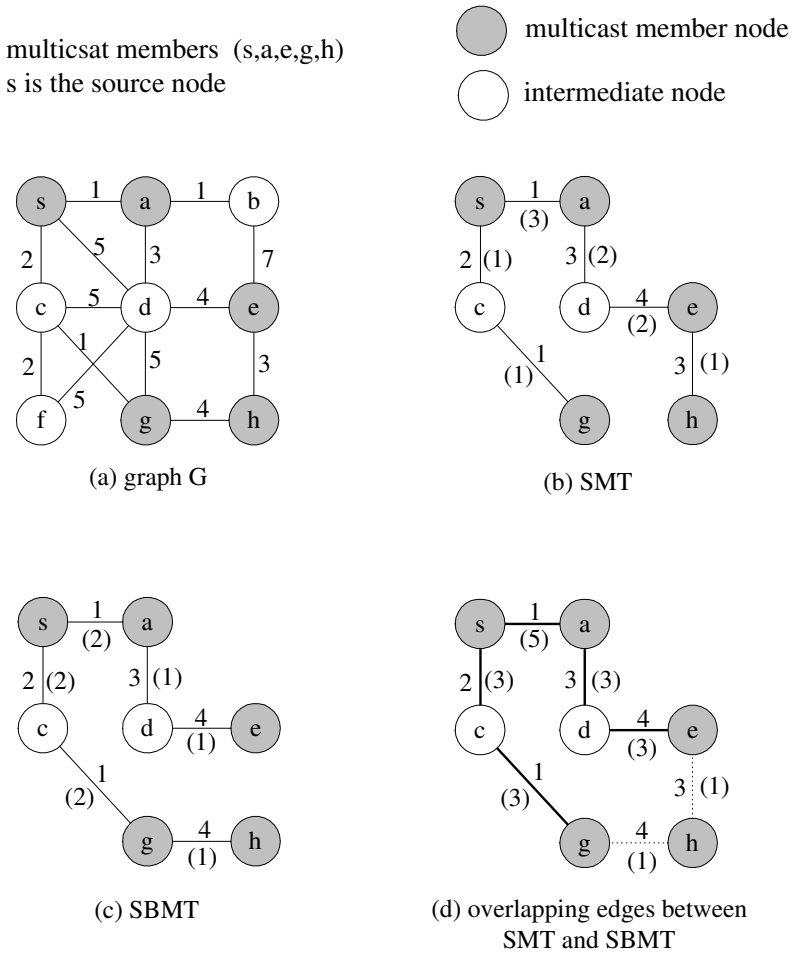


Figure 2: An example of overlapping edges between SMT and SBMT.

step. It is clear that the most 'critical' edge in all paths from source to every multicast member in a multicast tree should not fail. In other words, if it fails, many multicast members would no longer be able to receive packets sent from the source. Thus, we define a weighting function $g(e)$ to determine the importance of edge e in a multicast tree by calculating the number of paths passing through edge e among all paths from source to every destination node. Once again consider Fig. 2(b), for example. Since there are three paths, $s \xrightarrow{p} a$, $s \xrightarrow{p} e$ and $s \xrightarrow{p} h$, that pass through edge (s, a), the passing time of the edge (s, a) is three. So, we have $g_{SMT}(s, a) = 3$. Similarly, $g_{SMT}(a, d) = g_{SMT}(d, e) = 2$. Also we obtain $g_{SBMT}(s, a) = 2$ because of paths $s \xrightarrow{p} a$ and $s \xrightarrow{p} e$ as shown in Fig. 2(c). The number beneath each edge in the parentheses is the weight of the corresponding edge. Now, we can determine which overlapping edge is the proper candidate to replace. That is, the edge, say e, with the maximum value of $g_{SMT}(e) + g_{SBMT}(e)$ ($e \in T_{SMT}$ and $e \in T_{SBMT}$) will be selected. We denote

such an edge as a 'critical' edge for both SMT and SBMT. In Fig. 2(d), edge (s, a) is the critical edge.

2.3 The Analysis of Critical Edges

As described above, an edge e is selected for adjustment with the maximum value of $g_{SMT}(e)+g_{SBMT}(e)$. After replacing it, a new critical edge will be investigated and replaced, if any. Of course, there should exist lots of critical edges in the network and we need to adjust all of these critical edges in SBMT. In some situations, all overlapping edges may be replaced successfully. But the SBMT will waste bandwidth seriously as mentioned before. The key points are (i) how to define an edge as highly critical, and (ii) how to quantify the number of existing critical edges. Before describing this problem, some statistics terminology must first be addressed. Let μ and σ denote the mean and standard deviation of a set of data, respectively. The proportion of data falling within k standard deviations of the mean, which is called Chebyshev's Theorem [12], is given by $1 - 1/k^2$, where $k > 1$. The reason for employing Chebyshev's theorem to distinguish critical edges is that Chebyshev's theorem can be applied without information on the shape of a data set's distribution. That is, an edge e is a critical edge if it satisfies $g_{SMT}(e) + g_{SBMT}(e) \geq \mu + k\sigma$. It is obvious that the number of critical edges depends on k. Denote R as the ratio of the number of critical edges to the total number of edges in SMT. We have $R \cong 1/2k^2$ and it varies from 0 ($k \approx \infty$) to 0.5 ($k \approx 1$). In the case $R > 0.5$, the SMT would require that more than half of edges be protected by SBMT. This would be impractical and inefficient for a network with a low link failure probability. Let Fr denote the normalized reliability factor of SMT. We let $Fr = 2R$ (i.e., $0 < Fr < 1$) and the corresponding k will be roughly derived by the following formula:

$$k \approx \sqrt{\frac{1}{Fr}}. \tag{2}$$

Let CE denote the set that contains all critical edges in graph for a given k (from a given Fr). We have

$$CE = \{\, e \mid g_{SMT}(e) + g_{SBMT}(e) \geq \mu + k\sigma,\ e \in T_{SMT} \text{ and } e \in T_{SBMT}\}. \tag{3}$$

The procedure of finding a different route to replace each critical edge in SBMT is repeated until $CE = \phi$. We note that each time a critical edge is adjusted, set CE should be re-estimated.

3 Steiner Backup Multicast Tree (SBMT)

3.1 Fixed SMT Algorithm (FSA)

We now present the algorithms for deriving the SBMT. In this section, we propose two different algorithms: the fixed SMT algorithm (FSA) and the adaptive SMT algorithm (ASA).

In FSA, the total cost and topology of SMT are kept unchanged. All actions are focused on the SBMT. That is, the cost and the topology of the SBMT are changed dynamically during the procedure of adjustments. This algorithm employs the KMB heuristic algorithm to construct the T_{SMT} [7]. The basic concept of the KMB is briefly addressed as follows. Given a connected graph $G = (V, E)$ with a set $D \subseteq V$ containing the multicast members ($D = \{d_1, d_2, \cdots, d_{|D|}\}$), it constructs a complete graph $G' = (D, E')$ from G. For each edge $(v_i, v_j) \in E'$, the cost on it is assigned as the shortest path from v_i to v_j in G. A minimum cost spanning tree T' can be constructed from G' by using either Prime's or Kruskal's algorithm [11]. Finally, each edge in T' is replaced by its corresponding shortest path in G in order to form a solution, which is the SMT. It is obvious that, in each replacement operation, we must discard the edge in the path, which forms a loop in SMT.

The FSA will replace an edge of the minimum cost in SMT by its second-shortest path to form a SBMT. Let $SP = \{SP_{(u,v)}, \forall\ u, v \in D\}$ denote the set of shortest paths between any two vertices in D and $SSP = \{SSP_{(u,v)},\ \forall\ u, v \in V\}$ denote the set of the second shortest paths between any two vertices in V. The former is used to construct the temporary spanning tree from the complete graph G'. The latter is used to construct the original SBMT and to back up critical edges. The procedure of adjustment is repeated until there is no common edge in CE. As a result, the SBMT is found. We note that the shortest path between any two vertices in D and the second shortest path in V can be easily computed prior to executing this algorithm by modifying Dijkstra's algorithm [11]. We now formally address the FSA algorithm in Fig. 3.

3.2 Adaptive SMT Algorithm (ASA)

When given a large Fr (i.e. smaller k), more edges in SMT are protected by SBMT and the SBMT will have a large cost. In such a situation, the network utilization will be degraded significantly by reserving bulk bandwidth. Besides, after the SBMT recovers many failures, the multicast tree will become inefficient in terms of cost. This phenomenon caused by the network is quite unreliable. Therefore, we propose an adaptive SMT algorithm (ASA) to find a tolerable primary multicast tree and a better backup tree such that the cost difference between them is minimal. We note that such a criterion is quite different from establishing the minimum cost Steiner tree.

The ASA first finds the shortest path and the second shortest path from the source to each destination. Let node s denote the source node in D and $SP = \{SP_{(s,u)}, \forall\ u \in D - \{s\}\}$ and $SSP = \{SSP_{(s,u)}, \forall\ u \in D - \{s\}\}$ denote the sets of the shortest path and the second shortest path from source node s to each vertex in $D - \{s\}$, respectively. For simplicity, let $P = SP \bigcup SSP = \{P_1, P_2, \cdots, P_{2|D|-2}\}$ where $w(P_i) \leq w(P_j)$ if $i < j$. The ASA tries to separate set P into two subsets, say P_A and P_B, such that the cost difference between them is as small as possible (we note that P_A and P_B contain the edges in SMT and SBMT, respectively). For

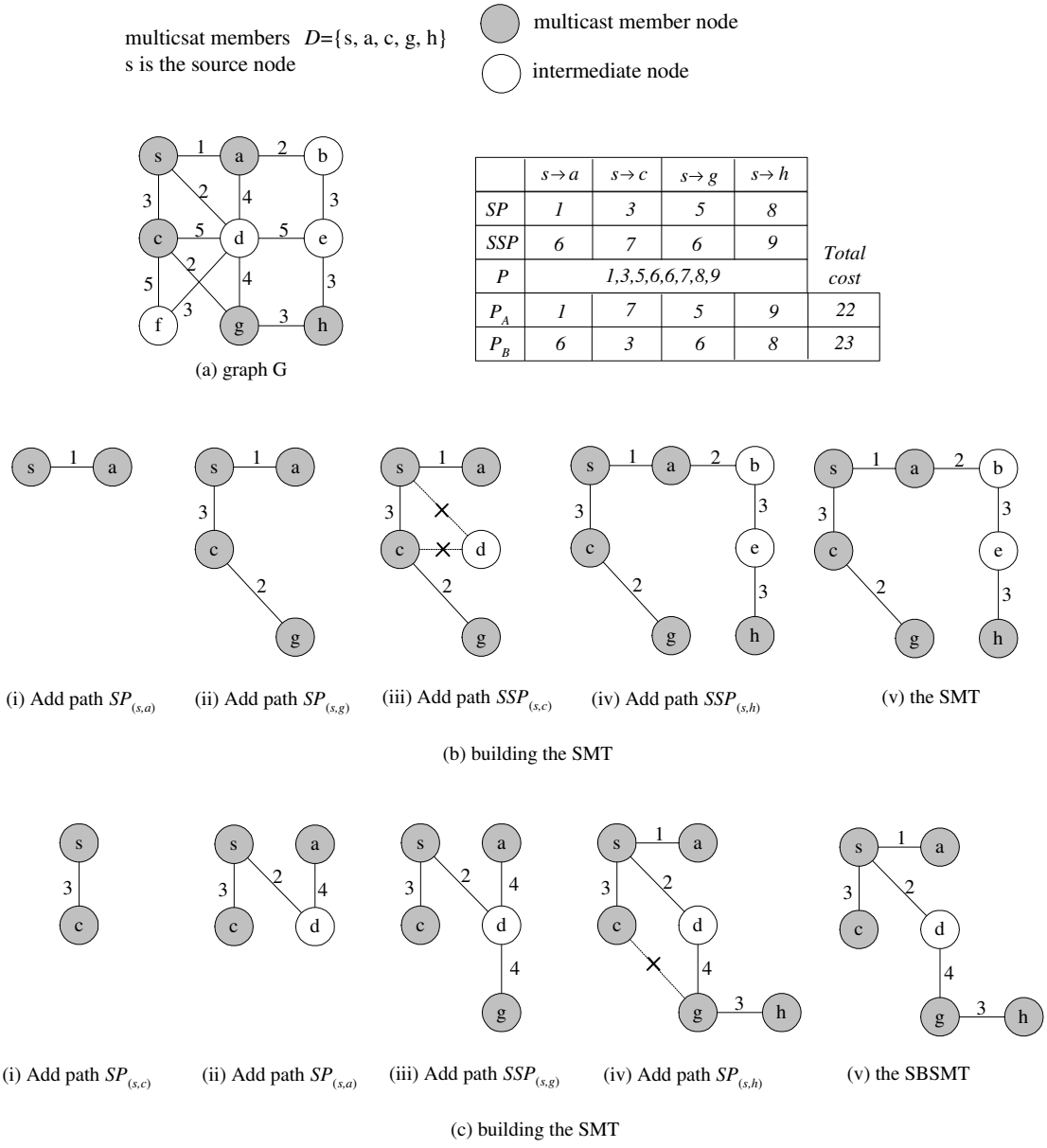
FIXED SMT ALGORITHM (FSA)
Input: an undirected graph $G = (V, E)$ with cost function $w : E \rightarrow Z^+$, a set $D \subseteq V$
Output: two trees T_{SMT} and T_{SBMT}; T_{SMT} is the minimum cost Steiner multicast tree and T_{SBMT} is a Steiner backup multicast tree
Employ the Dijkstra's algorithm to derive the SP and SSP sets
Construct the T_{SMT} by KMB algorithm
Choose an edge (p, q) from T_{SMT} of the lowest cost
$T_{SBMT} = T_{SMT} - \{(p, q)\} \cup SSP_{(p,q)}$ // replace one of SP by SSP
For each edge (u, v) in T_{SMT} and T_{SBMT}
  For each edge (r, s) in path $SP_{(u,v)}$
    If ((r, s) does not create a cycle in T_{SMT}) add (r, s) to T_{SMT}
    If ((r, s) does not create a cycle in T_{SBMT}) add (r, s) to T_{SBMT}
Find $CE = \{e | g_{SMT}(e) + g_{SBMT}(e) \geq \mu + k\sigma, e \in T_{SMT} and e \in T_{SBMT}\}$
While ($CE \neq \phi$)
  Find an edge e from CE such that $g_{SMT}(e) + g_{SBMT}(e)$ is maximal
  $T_{SBMT} = T_{SBMT} - \{e\}$
    For each edge (r, s) in path SSP_e
      If ((r, s) does not create a cycle in T_{SBMT})
        $T_{SBMT} = T_{SBMT} \cup SSP_{(r,s)}$
  $CE = CE - \{e\}$
Return T_{SMT} and T_{SBMT}

Figure 3: Fixed SMT Algorithm (FSA)

simplicity, let $Cost(S)$ denote the total cost of set S and is calculated as $Cost(S) = \sum_{P_i \in S} w(P_i)$. The ASA will minimize the difference between $Cost(P_A)$ and $Cost(P_B)$.

Actually, how to partition a set of numbers into two disjoint subsets so that the difference of the sums of two subsets is minimized has been shown to be NP-complete [13]. Hence, we propose a heuristic algorithm to solve this problem. Initially, let $P_A = P$ and $P_B = \phi$. When $Cost(P_A)$ is larger than $Cost(P_B)$, paths of the smallest cost in P_A are moved from set P_A to P_B one by one until the condition is no longer satisfied. Contrarily, if $Cost(P_A)$ is smaller than $Cost(P_B)$, a similar procedure is performed except that path(s) are moved from P_B to P_A. To prevent the algorithm from executing infinitely, a threshold $Epoch$ is employed to stop the algorithm. When either the cost difference is zero or the iteration reaches the threshold $Epoch$, the SMT and SBMT will be constructed from sets P_A and P_B, respectively. Two points must be noticed during construction of SMT and SBMT. One is that both $SP_{(s,u)}$ and $SSP_{(s,u)}$ for vertex u can not be located on the same tree because a loop would be formed. The other is that if a loop is formed when adding a new path, the new added edges along this loop should be discarded. An example of constructing SMT and SBMT is given in Fig. 4.

The graph G is a graph with 9 nodes. The cost of each link is given on the link shown in Fig. 4(a). Let node s be the source node and let the multicast members be the nodes a, c, g and h. First, the shortest path and the second shortest path from the source to each multicast member are found. The cost of each path is recorded. All eight paths are



	$s \to a$	$s \to c$	$s \to g$	$s \to h$	
SP	*1*	*3*	*5*	*8*	
SSP	*6*	*7*	*6*	*9*	*Total cost*
P	*1,3,5,6,6,7,8,9*				
P_A	*1*	*7*	*5*	*9*	*22*
P_B	*6*	*3*	*6*	*8*	*23*

Figure 4: An example of constructing the SMT and SBMT by adaptive SMT algorithm (ASA).

separated into two sets P_A and P_B according to the proposed strategy. We have two sets: $P_A = \{SP_{(s,a)}, SSP_{(s,c)}, SP_{(s,g)}, SSP_{(s,h)}\}$ and $P_B = \{SSP_{(s,a)}, SP_{(s,c)}, SSP_{(s,g)}, SP_{(s,h)}\}$. Moreover, $Cost(P_A) = 22$ and $Cost(P_B) = 23$. We note that if both $SP_{(s,u)}$ and $SSP_{(s,u)}$ are assigned in the same set, one of them must be moved to another set. Finally, each set will contain $|D| - 1$ different paths. Having both sets P_A and P_B, we may begin to build the SMT and SBMT, respectively. The procedure is illustrated in Figures 4(b) and 4(c). The shaded nodes in

ADAPTIVE SMT ALGORITHM (ASA)
Input: an undirected graph $G = (V, E)$ with cost function $w : E \rightarrow Z^+$, a set $D \subseteq V$ and a threshold *Epoch*
Output: two trees T_{SMT} and T_{SBMT}; T_{SMT} is the primary multicast tree and T_{SBMT} is a backup multicast tree such that the cost difference between them is minimal, i.e., minimize $Cost(T_{SBMT}) - Cost(T_{SMT})$
Employ the Dijkstra's algorithm to derive the *SP* and *SSP* sets
$P = SP \cup SSP; P_A = P; P_B = \phi$
$Loop = 0$
While ($Cost(P_A) \neq Cost(P_B)$ and $++Loop < Epoch$)
  if ($Cost(P_A) > Cost(P_B)$)
    Choose path p of the lowest cost from P_A
    $P_A = P_A - \{p\}$
    $P_B = P_B + \{P\}$
  else
    Choose path p of the lowest cost from P_B
    $P_B = P_B - \{p\}$
    $P_A = P_A + \{P\}$
For each vertex i where $SP_{(s,i)}$ and $SSP_{(s,i)}$ in a same set
  Move one of them into another set
For each path $P_{(s,v)}$ in P_A // construct the actual multicast tree T_{SMT}
  For each edge (r, s) in path $P_{(s,v)}$
    If ((r, s) does not create a cycle in T_SMT) add (r, s) to T_{SMT}
For each path $P_{(s,v)}$ in P_B // recover the actual backup multicast tree T_{SBMT}
  For each edge (r, s) in path $P_{(s,v)}$
    If ((r, s) does not create a cycle in T_{SBMT}) add (r, s) to T_{SBMT}
Backup each critical edge as lines 10–19 in FSA algorithm
Return T_{SMT} and T_{SBMT}

Figure 5: Adaptive SMT Algorithm (ASA)

Fig. 4 are the multicast members and node s is the source node. Recall that when adding the selected path to a multicast tree, edges along the added path that form a cycle will be removed. Consider Fig. 4(b), for example. When adding $SSP_{(s,c)}$, two edges (s, d) and (d, c) (shown by dotted line) are pruned from the SMT. Similarly, edge (c, g) is removed when adding $SP_{(s,h)}$ in Fig. 4(c). After adding all paths in sets P_A and P_B, both SMT and SBMT are constructed completely. Based on this method, there may still exist many overlapping edges. Therefore, the following adjustment step is required and repeated until there is no critical edge in *CE*. We now formally address the ASA algorithm in Fig. 5.

4 Dynamic Joining and Leaving in SMT and SBMT

4.1 Node Joining

The key point for node joining is how to find an optimal attachment point. Two methods, WGA [5] and shortest path (SP) [11], are proposed for finding the attachment node. In WGA,

suppose a node s is the owner of the multicast tree and remains in this tree. A node u is added to a multicast tree by choosing a node v in this tree which minimizes the function

$$w(v) = (1 - w) \times d(u, v) + w \times d(v, s), \tag{4}$$

where $0 \leq w \leq 0.5$ and $d(x, y)$ is the distance(cost) from x to y. In the SP, a shortest path from the source to the joined node is calculated. Since we have established a backup tree for the SMT, both SMT and SBMT can be used to speed up the process of finding the attachment point. There are four cases to be considered when a node is joined. For simplicity, let node u be the new joining node in the following cases. An example of dynamic joining with four cases is shown in Fig. 6.

Case 1: $u \in T_{SMT}$ and $u \in T_{SBMT}$

This is the simplest case where the added node is already in both trees. This added node is directly joined to both trees. Therefore, there is a unique path from source to this added node. The topology and total cost of both trees are unchanged. In Fig. 6(b), node d is directly joined to both trees.

Case 2: $u \in T_{SMT}$ and $u \notin T_{SBMT}$

The node u is joined directly to SMT as described in case 1. In order to add node u into SBMT, we attempt to find all paths in SMT from node u to the vertex which is also in SBMT. Let $P_n = \{P | P \subseteq u \xrightarrow{p} v,\ v \in T_{SMT} \text{ and } v \in T_{SBMT}\}$ denote the set of all available paths. We choose the shortest path among P_n and add it to SBMT. In Fig. 6(c), node c is directly joined to SMT and $P_n = \{c \xrightarrow{p} s, c \xrightarrow{p} g\}$. Because $c \xrightarrow{p} g$ is the shortest path, it is added to SBMT. We note that the path from s to c in SMT is different from that in SBMT.

Case 3: $u \notin T_{SMT}$ and $u \in T_{SBMT}$

Obviously, the method is the same as case 2. Node u is joined to SBMT directly. Then, we attempt to find the shortest path among P_n and migrate this path to SMT. In Fig. 6(d), b is the added node and $P_n = \{b \xrightarrow{p} a, b \xrightarrow{p} g, b \xrightarrow{p} e\}$, therefore, $b \xrightarrow{p} a$ is added to SMT. We note that the delay required for node joining in this case will be reduced significantly.

Case 4: $u \notin T_{SMT}$ and $u \notin T_{SBMT}$

First of all, the WGA is used to find the attachment node in both trees. Then, the path from node u to the attachment node is directly connected in both trees. In the case of adding node f in Fig. 6(e), path $c \xrightarrow{p} f$ is placed in SBMT while $d \xrightarrow{p} f$ is placed in SMT.

4.2 Node Leave

Basically, we only consider the leaving node as an internal node or a leaf node for both trees. If it is a leaf node, it is deleted and the ancestors of the node are repeatedly pruned until a destination node or source node is reached. Otherwise, this node is just marked as intermediate node only.

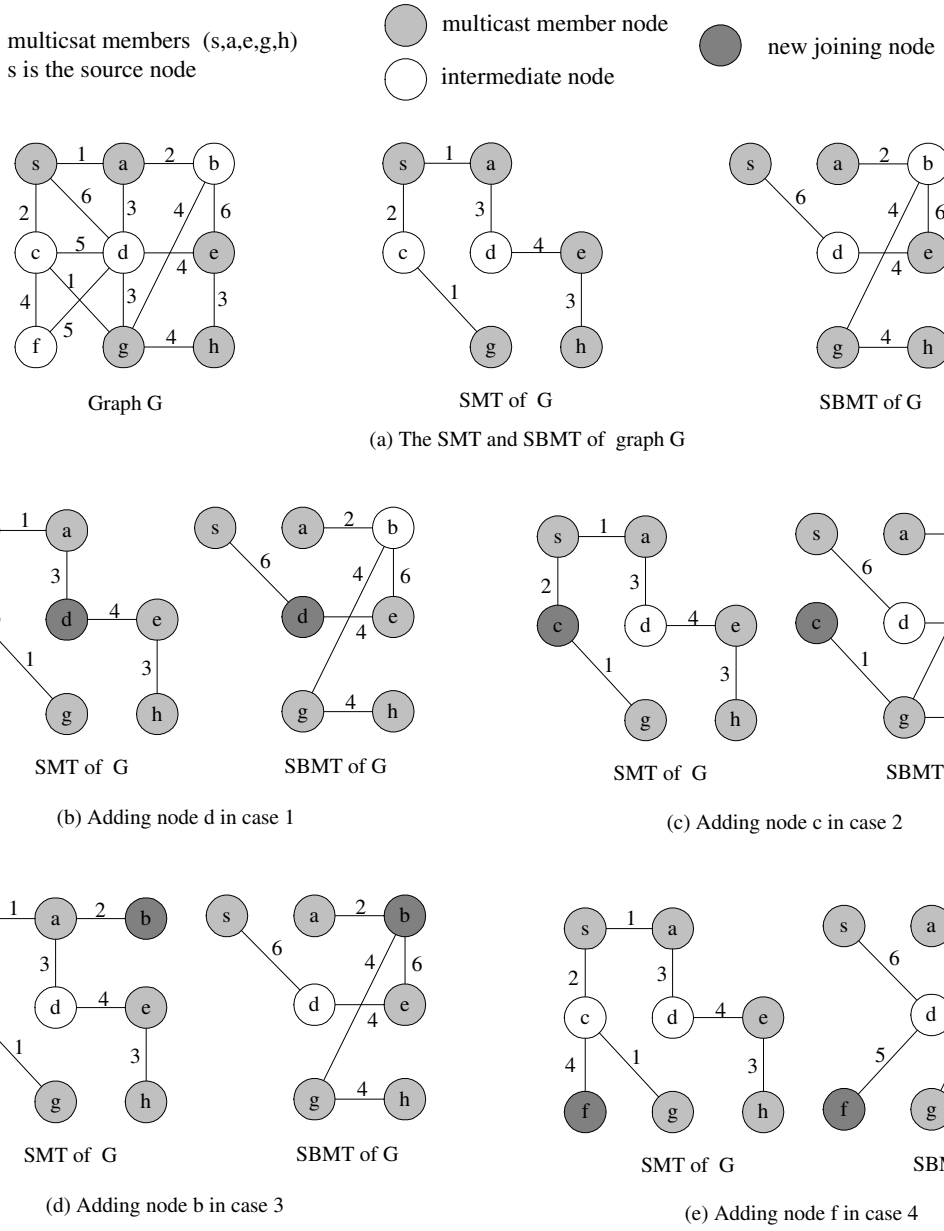


Figure 6: Examples of node dynamic joining.

5 Simulation Model and Results

5.1 Simulation Model

In our experiment, we run two network models, a sparse network model (SNM) and a dense network model (DNM). We say that a network belongs to SNM if $|E| \ll |V|^2$. On the other hand, a network belongs to DNM if $|E| \cong |V|^2$ [11]. Both network models are similar to that of

[5]. Nodes are randomly distributed in a rectangular coordinate grid. Each node is located using integer coordinates. The distance is chosen for each pair of nodes in $(0, L]$ from an uniform random distribution. Each node in the simulation represents a switch or router, which has some of the characteristics of an actual network. Also, each edge represents a bi-directional physical link and is generated between pairs of nodes u, v with a probability that depends on the distance between them. The edge probability is given by [5]

$$p(\{u, v\}) = \beta \exp \frac{-d(u, v)}{L\alpha}, \tag{5}$$

where $d(u, v)$ is the distance from node u to v, L is the maximum distance between two nodes, and α and β are parameters in the range $(0, 1]$ and control the characteristics of the graph produced. Increasing β will increase the average vertex degree of the graph. Increasing α will increase the ratio of longer edges relative to shorter ones. Finally, the cost of each edge is set to the distance between its endpoints. In our simulations, a SNM is obtained when α and β are set to 0.2 and 0.25, respectively while a DNM is obtained if both α and β are set to 0.5. For both models, the maximal distance L between adjacent nodes is set to 50. For each data point in the simulation results, 20 random graphs are generated for each model. For each graph of the respective model, the corresponding multicast tree is established and measured. Each reported data point is calculated as the average of the 20 collected data points.

The performance of the proposed SBMT is evaluated in terms of the *unreliable factor* (Fu) which is defined as follows:

$$Fu = \frac{OE}{E} \tag{6}$$

where OE is the number of overlapping edges between SMT and SBMT and E is the number of edges in SMT. The larger the Fu is, the less reliability the network has.

For investigating the efficiency of dynamic joining, a sequence of 1000 requests of adding or removing nodes is monitored. The probability that a node adds to the multicast group or removes from a multicast tree is determined by the probability function PC [5] which is defined as follows

$$P_c = \frac{\alpha(n-k)}{\alpha(n-k) + (1-\alpha)k}, \tag{7}$$

where k is the current number of nodes in multicast group, n is the number of nodes in network, and α is a real-number parameter in the range $(0, 1)$. The hit ratio Hr is defined as the number of nodes added, which are just located within either SMT or SBMT, to the total number of joining requests.

5.2 Simulation Results and Discussion

The performance of both models was compared by tree costs, Fu, and Hr. First of all, we study the costs of SMT and SBMT constructed by the FSA and ASA for different sizes of

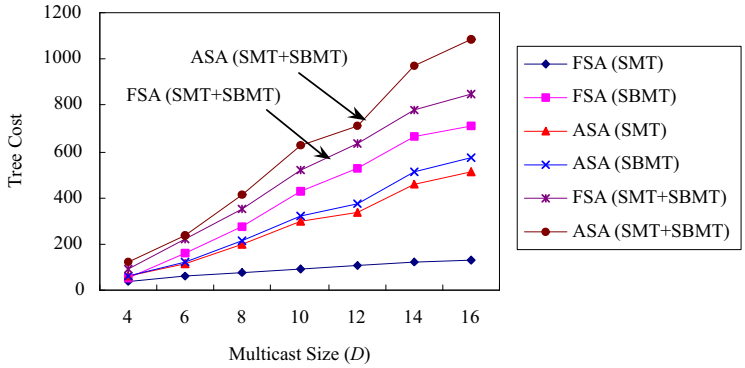


Figure 7: Comparisons of tree costs obtained by FSA and ASA based on DNM under different multicast sizes.

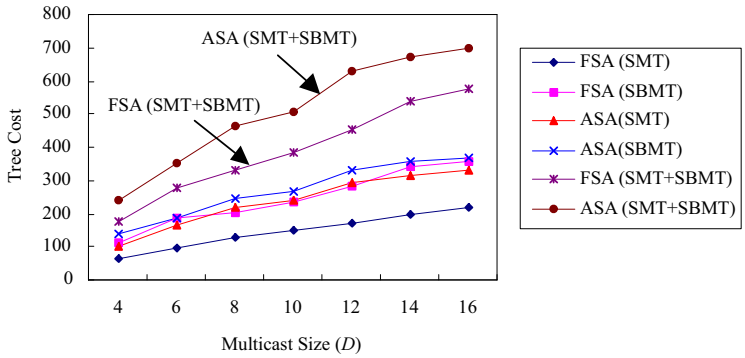


Figure 8: Comparisons of tree costs obtained by FSA and ASA based on SNM under different multicast sizes.

multicast group D in a graph with 100 nodes. The multicast size D is considered from 4 to 16 in steps of 2. Each subset D contains nodes selected at random from the 100 nodes. For each algorithm, two curves are plotted for the tree costs of SMT and SBMT, respectively. The tree costs obtained by different algorithms for DNM and SNM were shown in Figures 7 and 8, respectively. In Fig. 7, it is obvious that the FSA has the smallest *Cost*(SMT) and the largest *Cost*(SBMT), respectively. Also, the cost difference between SMT and SBMT (i.e., *Cost*(SBMT)−*Cost*(SMT)) is dependent on the multicast size. We observe that the larger the multicast size is, the larger the cost difference is.

Contrarily, the cost difference between SMT and SBMT derived by ASA is minimal for all cases. Therefore, when most of links fail in the network, the employed multicast tree SMT will be migrated into the SBMT and the total cost of tree will become very high if FSA is used. On the other hand, the cost of tree is still almost the same as the primary multicast tree if ASA is used. This implies that the proposed FSA and ASA are suitable for stable and unstable networks, respectively. However, the total cost *Cost*(SMT)+*Cost*(SBMT) of ASA is larger than

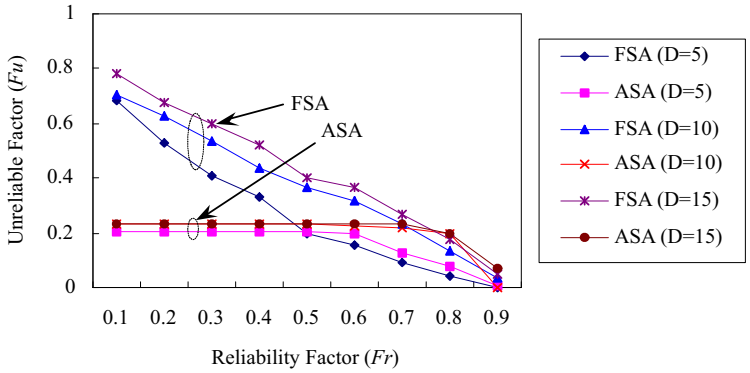


Figure 9: Comparisons of *Fu* obtained by FSA and ASA based DNM under different *D* and *Fr*.

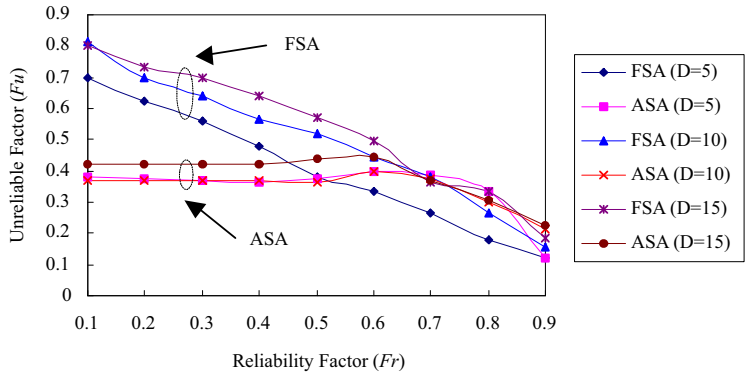


Figure 10: Comparisons of *Fu* obtained by FSA and ASA based SNM under different *D* and *Fr*.

that of FSA. Therefore, the timing of employing FSA or ASA is strongly dependent on the link reliability of an actual network. When a user desires more than half of the edges to be protected, the ASA algorithm should be applied to maintain a more reliable multicast tree when networks incur fatal failures on many links. Fig. 8 illustrates the tree costs obtained by FSA and ASA based on SNM under different multicast sizes. We can easily find that the FSA still has the smallest *Cost*(SMT) and the largest *Cost*(SBMT). Also, the *Cost*(SBMT) of FSA is smaller than *Cost*(SMT) of ASA when multicast sizes 8, 10, and 12 are considered. These are quite reasonable because the total number of *bridges* in a tree constructed by ASA is smaller than that constructed of by FSA. In other words, the number of overlapping edges between SMT and SBMT constructed by ASA is less than that obtained when SMT and SBMT are constructed by FSA.

The *Fu* of SMT was shown in Figures 9 and 10. The value of *reliability factor Fr* ranges from 0.1 to 0.9. Intuitively, as *Fr* increases, the *Fu* of SMT decreases. Fig. 9 based on the DNM shows that the *Fu* of the ASA is smaller than that of FSA except for the cases $Fr \geq 0.5$ where

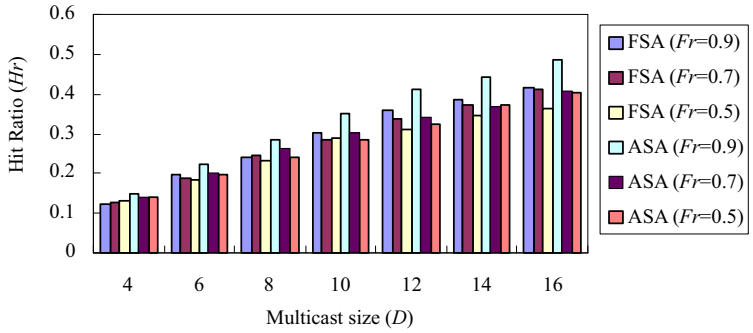


Figure 11: Comparisons of *Hr* of node dynamic joining obtained by FSA and ASA based on DNM under different *D* and *Fr*.

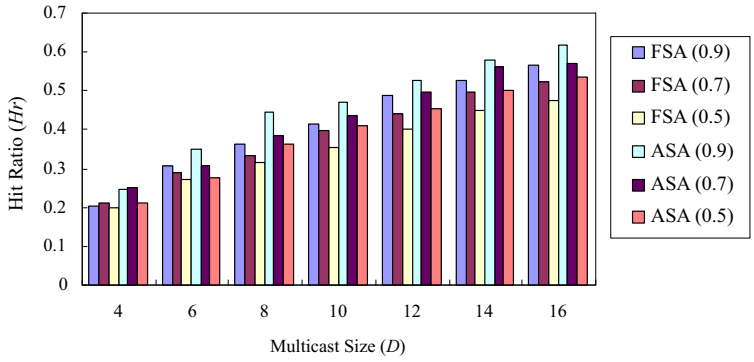


Figure 12: Comparisons of *Hr* of node dynamic joining obtained by FSA and ASA based on SNM under different *D* and *Fr*.

$D = 5$ and $Fr \geq 0.8$ where $D = 10$ and 15. We can also observe that the smaller the multicast size is, the smaller the *Fu* will be derived in both algorithms. That is, the overlapping edges are small for small multicast size. In general, the ASA algorithm can yield well results when $Fr \leq 0.6$. At this time, the *Fr*s are bounded within about 0.23. This means that we plan to establish a multicast service over an un-safety network by ASA, 77% link failure would not only be tolerated but the cost of multicast tree would almost have the same cost as the original one. This is another important fact to consider for employing the ASA algorithm.

For SNM shown in Fig. 10, the *Fu* of the ASA is smaller than that of FSA except for the cases $Fr \geq 0.6$ where $D = 5$ and $Fr \geq 0.8$ where $D = 5$ and 10. Also, even when $Fr = 0.9$, it is impossible to find a completely independent backup tree for SMT. The reason is that there are too many bridges in SNM. In general, the ASA algorithm can yield good results when $Fr \leq 0.4$. At this time, the *Fu*s are bounded within about 0.42. Obviously, the results are not better than that of DNM. The key differences are that *Fr* is smaller and only 58% link failure are tolerated.

The efficiency of dynamic joining with SBMT was shown in Figures 11 and 12 for DNM

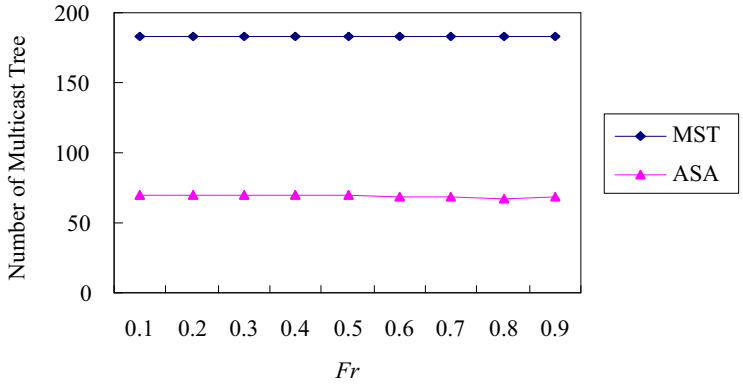


Figure 13: Comparisons of total number of multicast tree constructed by MST and ASA based on DNM under different Fr.

and SNM, respectively. The ASA has the higher hit ratio HR than the FSA. This is because the total number of nodes in SMT and SBMT in ASA is larger than that of FSA. As a result, the probability of adding a node just located in either SMT or SBMT is higher. In Fig. 11, we find that about 40% joining processes are improved when $D = 16$. We also found that when a large Fr is specified, a higher HR is obtained. This is because a high reliability factor will establish a 'bigger' SBMT. The maximal HR occurs when $Fr = 0.9$.

On the other hand, The ASA still has a the higher hit ratio HR than the FSA as shown in Fig. 12. This is because the total number of bridges in SMT and SBMT in ASA is less than that of FSA. As a result, the cost of the tree constructed by ASA is higher than that of the tree constructed by FSA (as described above). In Fig. 12, about 50% of the joining processes are improved when $D = 16$ since the total number of nodes in SMT and SBMT are larger than that of DNM. The maximal HR still occurs when $Fr = 0.9$.

The total number of SMTs established for different group based on DNM and SNM is shown in Figures 13 and 14. For both Figures, we used the minimal spanning tree (MST) to construct all SMTs. Since there are no SBMTs, the total number of SMTs established is independent of the Fr. On the other hand, ASA was used to construct both SMT and SBMT. Since Fr affects the size of k, the total number of SMTs is dependent on the Fr. Therefore, it is obvious that the MST can construct more SMTs than ASA no matter how much bandwidth is assigned to for each edge. Note that the difference in the of total number of SMTs derived by MST and ASA, respectively, in SNM is small. The reason is there are too many bridges in the original graph.

The channel utilization (CU) was shown in Figures 15 and 16. We first define the channel reservation (CR) as the total bandwidth reserved in advance and denote it as follows:

$$CR = \frac{TE_{SBMT} \times E_{Bandwidth}}{E_G \times E_{Bandwidth}}, \tag{8}$$

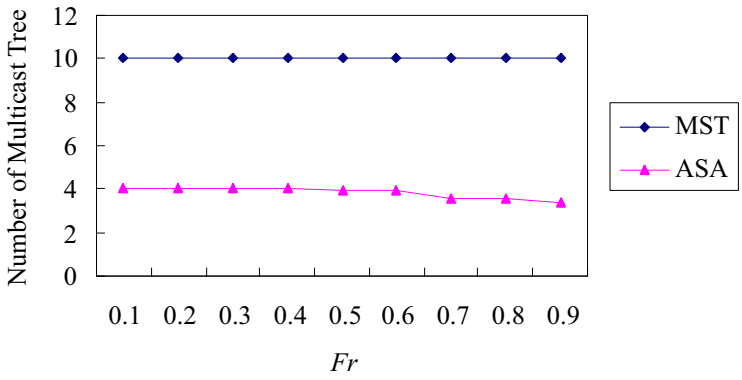


Figure 14: Comparisons of total number of multicast tree constructed by MST and ASA based on SNM under different Fr.

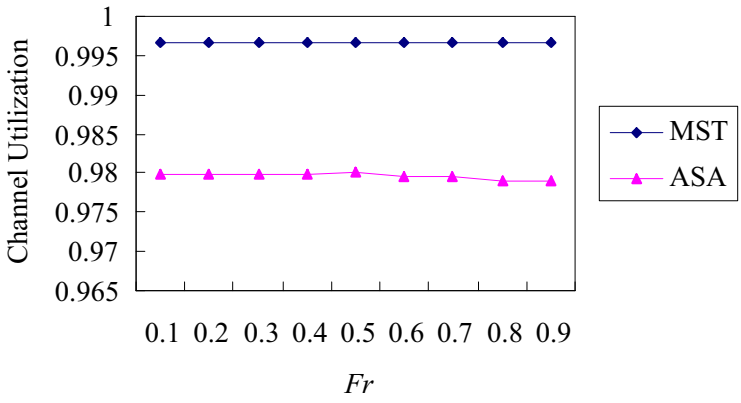


Figure 15: Comparisons of CU constructed by MST and ASA based on DNM under different Fr.

where TE_{SBMT} and E_G denote the total number of edges in SBMT and G, respectively. $E_{Bandwidth}$ denotes the bandwidth assigned to each edge. Similarly, the CU for each established SMT can be computed as follows:

$$CU = \frac{TE_{SMT} \times E_{Bandwidth}}{E_G \times E_{Bandwidth}}, \tag{9}$$

where TE_{SMT} denotes the total number of edges in SMT. Suppose the total number of SMT derived by MST is MT and total number of SMT and SBMT derived by ASA is SST. The total CU for MST and ASA can be computed as follows:

$$CU_{MST} = \sum_{i=1}^{MT} CU_i \tag{10}$$

$$CU_{ASA} = \sum_{i=1}^{SST} CU_i \tag{11}$$

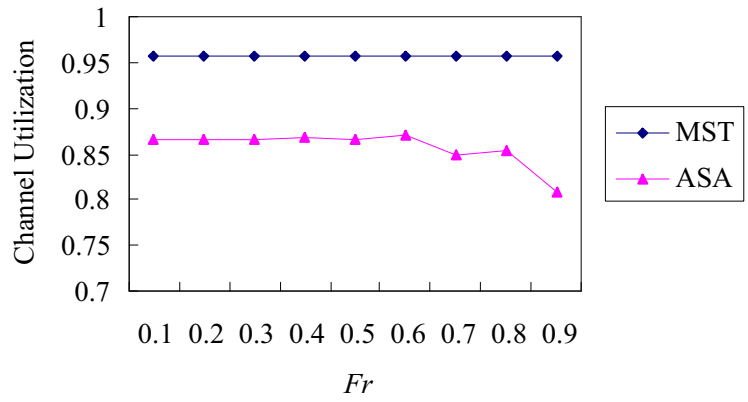


Figure 16: Comparisons of *CU* constructed by MST and ASA based on SNM under different *Fr*.

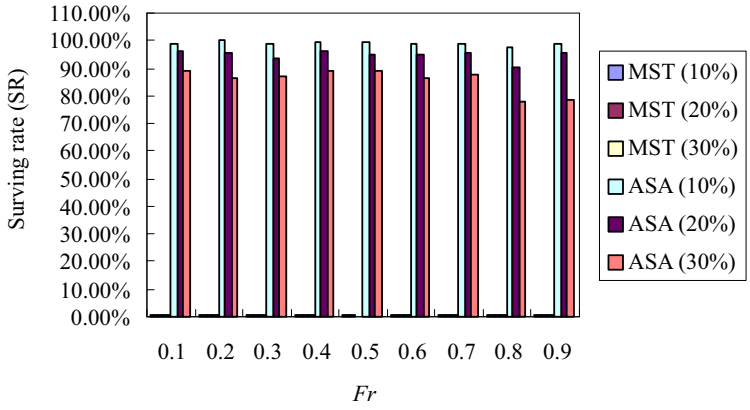


Figure 17: Comparisons of *SR* constructed by MST and ASA based on DNM under different *Fr*.

Obviously, from Figures 13 and 14, $MT > SST$. Thus, $CU_{MST} > CU_{ASA}$. Also, the CU_{MST} is independent of *Fr* while CU_{ASA} is dependent of the *Fr*. That is, the larger the *Fr* was, the smaller the *CU* is.

The surviving rate *SR* for DNM and SNM was shown in Figures 17 and 18, respectively. We suppose a total of 10%, 20%, and 30% edges of SMT are derived by MST and ASA as a result of failures in the network. The *SR* can be computed as follow:

$$SR = \frac{S_{member}}{|D|}, \tag{12}$$

where S_{member} denotes the total number of member nodes which remain in the SMT when these edges fail. Obviously, ASA had higher *SR* than that of MST. In our experiment, almost all failed edges can be backed up by SBMT. Contrarily, the *SR* of MST is very low since there is not enough bandwidth to find another path to backup a failed edge. However, the value of

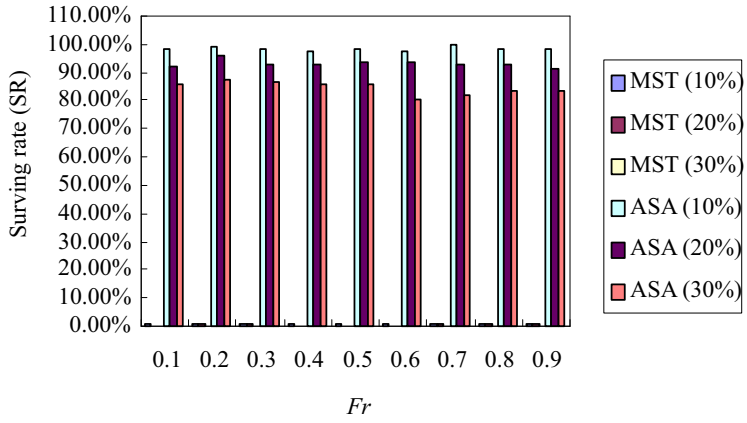


Figure 18: Comparisons of *SR* constructed by MST and ASA based on SNM under different *Fr*.

CU_{ASA} as shown in Figures 15 and 16 indicates that bandwidth must be reserved in advance for backup. This will waste much bandwidth.

6 Conclusion and Remarks

In this paper, two algorithms for the Steiner backup multicast tree (SBMT) problem are derived to find the SBMT in any network: the fixed SMT algorithm (FSA) and the adaptive SMT algorithm (ASA). The algorithms are based on our observation that it is important to protect the most critical overlapping edges between SMT and SBMT in order to obtain a highly reliable network. To this end,we adopted a statistic-based reliability factor to classify the critical edges in SMT. Simulation results show that the FSA and ASA are suitable for stable networks and unstable networks, respectively. Moreover, when adding a node into multicast group, both SMT and SBMT are used to find the best attachment node. Simulation results show that the proposed SBMT provides a significant improvement on the node-joining processes. Most of these simulation results remain the same when the network model is changed. Thus, it is clear that our algorithms can be applied to any network model.

7 Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable comments and Dr. Monticone for his kind assistance of editing this manuscript.

References

[1] L. Zhang, S. Deering, D. Estrin, S. Shenker, & D. Zappala, RSVP: A new resource reservation protocol, *IEEE Network Mag.*, 7(5), 1993, 8–18.

[2] S. Blake et al., An architecture for differentiated services, RFC 2475, 1998.

[3] P. Winter, Steiner problem in networks: A survey, *Networks*, 17, 1987, 129–167.

[4] R.M. Karp, *Reducibility Among Combinatorial Problems–Complexity of Computer Computations*, in Miller and Thatcher (Ed.), (New York: Plenum Press, 1972), 85–103.

[5] B.M. Waxman, Routing of multipoint connections, *IEEE J. Select. Areas Commun.*, 6(9), 1988, 1617–1622.

[6] H. Takahashi & A. Matsuyama, An approximate solution for the Steiner problem in graphs, *Math. Japonica*, 24, 1980, 573–577.

[7] L. Kou, G. Markowsky, & L. Berman, A fast algorithm for Steiner trees, *Acta Informatica*, 15, 1981, 141–145.

[8] R. Dighe, Q. Ren & B. Sengupta, A link based alternative routing scheme for network restoration under failure, *Proc. IEEE INFOCOM'95*, Boston, MA, 1995, 2118–2123.

[9] J. Anderson, B. Doshi, S. Dravida, & P. Harshavardhana, Fast restoration of ATM networks, *IEEE J. Select. Areas Commun.*, 12(1), 1994, 128–138.

[10] H. Sakauchi, Y. Nishimura, & S. Hasegawa, A self-healing network with an economical spare-channel assignment, *Proc. IEEE GLOBECOM'90*, 1990, 438–443.

[11] T.H. Cormen, C.E. Leiserson, & R.L. Rivest, *Introduction to Algorithms* (Cambridge, MA: MIT Press, 1990).

[12] D.R. Byrkit, *Statistics today: A comprehensive introduction* (Menlo Park, CA, 1987).

[13] M.R. Garey & D.S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness* (New York: W.H. Freeman and Company, 1979).

8 Biography

Wu-Hsiao Hsu received the Ph.D degree in Department of Computer Science and Information Engineering from Tamkang University, Taipei county, Taiwan in 1999. He is currently an assistant professor in department of computer science and information engineering at Ming-Chuan university, Taoyuan Country, Taiwan. His research interests are in QoS unicast/multicast routing, communication protocols and wireless networks.

Jenhui Chen received his B.S. and Ph.D. degree in Computer Science and Information Engineering (CSIE) from Tamkang University in 1998 and 2003, respectively. In the spring of 2003, he joined the faculty of Computer Science and Information Engineering Department at Chang Gung University and served as the Assistant Professor. He also occupies the supervisor of Network Department in the Information Center of Chang Gung University. Dr. Chen once served the reviewer of *IEEE Transactions on Wireless Communications*, *ACM/Kluwer Mobile Networks and Applications (MONET)*, *Journal of Information Science and Engineering*, and *International Journal of Electrical Engineering*. His main research interests include design, analysis, and implementation of communication and network protocols, wireless networks, milibots, and artificial intelligence. He is a member of ACM, IEEE, and IEICE.

Shiann-Tsong Sheu received his B.S. degree in Applied Mathematics from National Chung Hsing University in 1990, and obtained his Ph.D. degree in Computer Science from National Tsing Hua

University in May of 1995. From 1995 to 2002, he was an Associate Professor at the Department of Electrical Engineering, Tamkang University. Since February 2002, he has become a Professor at the Department of Electrical Engineering, Tamkang University. Dr. Sheu received the outstanding young researcher award by the IEEE Communication Society Asia Pacific Board in 2002. His research interests include next-generation wireless communication, WDM networks and intelligent control algorithms.

Chih-Feng Chao received the B.Eng. degree from the Department of Electrical Engineering, Chinese Culture University, Taiwan, in 1998. He worked as the chief in the Network Division of Information Technology Center in Kainan University, Taiwan, from 2000 to 2003. Currently, he is a Ph.D. candidate in the Department of Computer Science and Information Engineering, Tamkang University, Taiwan. His major research interests include mobile computing, wireless ad hoc and sensor networks, and multicast networking.